

CASE STUDY

MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE

ONLINE BANKING SYSTEM IN C :

Certainly! Here’s a detailed explanation of each part of the provided C program, which builds a basic bank account system using file handling.

1. Structure Definition

The core of the program revolves around the `Account` structure, which is designed to encapsulate all necessary information about a bank account. This structure includes:

- `accountNumber`: An integer representing the unique identifier for each account.

- `name`: A string to hold the account holder's name.

- `balance`: A float that stores the current balance of the account.

This structure allows the program to efficiently manage and manipulate account data.

2. Function Implementations

Account Creation (`account()`)

The `account()` function is responsible for creating a new account. It prompts the user for the account number, name, and initial balance. After collecting this information, the function opens a file in append binary mode (`"ab"`), which ensures that new accounts are added to the end of the file without overwriting existing data. The account details are then written to the file using `fwrite()`. If the file cannot be opened, an error message is displayed. This function plays a crucial role in setting up initial accounts for the system.

Money Transfer (`transferMoney()`)

The `transferMoney()` function facilitates the transfer of funds between two accounts. It first prompts the user to enter the source and destination account numbers and the amount to transfer. The function then loads the details of both accounts from the file using `loadAccount()`. If either account does not exist or the source account does not have sufficient funds, appropriate error messages are shown. If the transfer is valid, the balances of both accounts are updated accordingly. The updated account details are then saved back to the file using `saveAccount()`. This function ensures the integrity of account balances during transactions.

#### \*\*Balance Check (`checkBalance()`)\*\*

The `checkBalance()` function allows users to check the balance of a specific account. The user is prompted to enter the account number, and the function retrieves the account details from the file using `loadAccount()`. If the account exists, the function displays the account holder's name and balance. If the account cannot be found, an error message is displayed. This function provides a way for users to view their current financial status.

#### \*\*Login Simulation (`login()`)\*\*

The `login()` function simulates a login process by allowing users to view their account details based on their account number. The user enters their account number, and the function loads the corresponding account from the file using `loadAccount()`. If the account exists, the function displays the account holder's name and balance. This function provides a basic interface for users to verify their account details and balance.

### 3. \*\*File Handling\*\*

File handling is critical in this program as it manages the persistence of account data.

#### \*\*Saving Accounts (`saveAccount()`)\*\*

The `saveAccount()` function updates the file with the latest account information. It creates a temporary file (`"temp.dat"`) and writes all existing accounts to it, replacing the account with the updated information if it matches the account being saved. Once all accounts are processed, the original file is deleted and the temporary file is renamed to replace it. This method ensures that account updates are correctly recorded.

#### \*\*Loading Accounts (`loadAccount()`)\*\*

The `loadAccount()` function retrieves account details from the file based on the account number provided. It opens the file in binary read mode (`"rb"`) and searches through the records. If the account with the specified number is found, its details are returned; otherwise, the function returns `NULL`. This function is essential for reading and manipulating account data.

### 4. \*\*Menu System\*\*

The `main()` function provides a simple menu-driven interface for users to interact with the system. It repeatedly displays a menu with options to create an account, transfer money, check balance, or log in. Based on the user's choice, the corresponding function is called using a `switch` statement. This loop continues until the user chooses to exit the program.

Summary

This C program provides a fundamental banking system using file handling to manage account data. By defining a structure to hold account details, implementing functions for various operations, and using file handling to persist data, the program offers a basic yet functional example of a bank account management system. The menu system allows users to interact with the program and perform tasks such as account creation, money transfers, balance checks, and logging in. This implementation demonstrates core concepts in C programming, including structures, file handling, and user interfaces.

GitHub Repository Website Link:

<https://github.com/Vishaldurairaj/online_banking.git>

Screenshots of the project:

A black screen with a black border

Description automatically generated

A black screen with a black border

Description automatically generated